

AD-A079 415

GENERAL ELECTRIC CO SYRACUSE N Y HEAVY MILITARY EQUI--ETC F/G 9/2
THE SOFTWARE DEVELOPMENT NOTEBOOK.(U)
NOV 79 J MCKISSICK , R A PRICE

UNCLASSIFIED

NL

1 1
ALIA
0764-0



END
DATE
FILMED

2 -8C

DDC

TIS Distribution Center
CSP 4-18, X7712
Syracuse, New York 13221

LEVEL

GENERAL ELECTRIC

HEAVY MILITARY EQUIPMENT DEPARTMENT

DDC
FORM

JAN 15 1980

9 TECHNICAL INFORMATION SERIES

10 Author J. McKissick R.A./Price		Subject Category Software Development		No. R79EHM11	
				11 Date Nov 79	
Title 6 The Software Development Notebook.					
Copies Available at HMED TIS Distribution Center Box 1122 (CSP 4-18) Syracuse, New York 13221			GE Class 1		No. of Pages 8
			Govt Class U		
Summary A Software Development Notebook (SDN) technique is documented in this paper. The organization, content, use and audit of SDNs are described. Experience and results from the application of this technique are also presented. The paper was prepared for the 1979 Annual Reliability and Maintainability Symposium. Its publication as a TIS is at the request of the Aerospace Business Group's Software Engineering Panel.					
Key Words: Software, Software Audit, Software Maintainability, Software Quality, Software Reliability					

This document has been approved for public release and sale; its distribution is unlimited.

This document contains proprietary information of the General Electric Company and is restricted to distribution and use within the General Electric Company unless designated above as GE Class 1 or unless otherwise expressly authorized in writing.

ADA 079415

DDC FILE COPY

408 969 mt

80 1-10 1146

Send to _____

GENERAL ELECTRIC COMPANY TECHNICAL INFORMATION

Within the limitations imposed by Government data export regulations and security classifications, the availability of General Electric Company technical information is regulated by the following classifications in order to safeguard proprietary information:

CLASS 1: GENERAL INFORMATION

Available to anyone on request.
Patent, legal and commercial review
required before issue.

CLASS 2: GENERAL COMPANY INFORMATION

Available to any General Electric Company
employee on request.
Available to any General Electric Subsidiary
or Licensee subject to existing agreements.
Disclosure outside General Electric Company
requires approval of originating component.

CLASS 3: LIMITED AVAILABILITY INFORMATION

Original Distribution to those individuals with
specific need for information.
Subsequent Company availability requires
originating component approval.
Disclosure outside General Electric Company
requires approval of originating component.

CLASS 4: HIGHLY RESTRICTED DISTRIBUTION

Original distribution to those individuals personally responsible for the Company's interests in the subject.
Copies serially numbered, assigned and recorded by name.
Material content, and knowledge of existence, restricted to copy holder.

GOVERNMENT SECURITY CLASSIFICATIONS, when required, take precedence in the handling of the material. Wherever not specifically disallowed, the General Electric classifications should also be included in order to obtain proper handling routines.

GENERAL ELECTRIC COMPANY
HEAVY MILITARY EQUIPMENT DEPARTMENT
TECHNICAL INFORMATION SERIES

SECTION Reliability and Quality Assurance
UNIT Computer Software R&QA
HMED ACCOUNTING REFERENCE _____
COLLABORATORS J. McKissick, R.A. Price
APPROVED _____ TITLE _____ LOCATION _____

MINIMUM DISTRIBUTION - Government Unclassified Material (and Title Pages) in G.E. Classes 1, 2, or 3 will be the following.

<u>Copies</u>	<u>Title Page Only</u>	<u>To</u>
0	1	Legal Section, HMED (Syracuse)
0	1	Manager, Technological Planning, HMED (Syracuse)
5	6	G-E Technical Data Center (Schenectady)

MINIMUM DISTRIBUTION - Government Classified Material, Secret or Confidential in G.E. Classes 1, 2, or 3 will be the following.

1	1	Classified Section, Electronics Park Library
1	0	Manager, Technological Planning, HMED (Syracuse)

ADDITIONAL DISTRIBUTION (Keep at minimum within intent of assigned G.E. Class.)

<u>COPIES</u>	<u>NAME</u>	<u>LOCATION</u>
5	DEFENSE DOCUMENTATION	CAMERON STATION,
(CLASS 1 ONLY)	CENTER	ALEXANDRIA, VA. 22314
1	L. I. Chasen	P.O. Box 8555
		Philadelphia, Pa., 19101

See attached list.

DISTRIBUTION LIST

Advanced Development Council

A.P. Belle Isle	H. Lehmann
L.E. Foster	P.C. Luttenberger
C.H. Hunt	S. Matt
M. Hunter	J.W. McKee
M. Isaac	F.W. Mezger
M.A. Johnson	W.H. Raber
K.M. Kain	T.A. Shaw
R.A. Kashnow	V. Stout
H.J. Kindl	J.J. Suran

Reliability Assurance Panel

P. Albrecht	B.D. Kringelman
A.E. Allen	D.J. LaCombe
J.G. Beasley	R.E. Ledbetter
E.J. Benman	W.J. MacFarland
M.L. Bienvenu	E.E. MacMorland
W.T. Chapin	A.L. Mannino
C.R. Church	R. McCandless
G.E. Desaulniers	T.D. McCormick
R. Freedman	E. Meese
W.H. Goddeau	M.H. Monahan
F. Hausner	V.A. Musick
C.R. Heising	J. Olsson
A.E. Herr	A.M. Smith
K. Hoffmockel	W.M. Starr
T.H. Hyatt	L.D. Test
F.W. Kielher	R.E. Warr
W. Kindig	P. Watson

Software Engineering Panel

R. Allen	G. Leonard
R. Amstutz	C. Migliarina
R. Breuning	L. Oliver
J. Dash	D. Panzel
S. Fix	G. Walters
J. Geyer	R. Wakefield
N. Hallquist	A. Williams
R. Jaffe	J.W. Wood, Jr.
L. Lambert	

Software R&QA

R. Baessler	J. McCall
F. Basmajian	P. Milliman
J. Brunner	D. Nelson
A.D. Buehler	N. Pelletier
J. Callahan	D. Pepin
P. Castiglione	W. Pollard
T. Crumley	A. Putash
Dr. W. Curtis	Z. Quastler
H. Dozier	R. Raymond
J. Fallon	G. Reagan
A. Garawitz	P. Richards
J. Gardner	D. Schettler
J. Griffith	R. Schneider
M. Halvorsen	R. Servadio
W. Junk	R. Shuey
G. Larson	R. W. Signor
M. Laymon	R. Sperling
T. Love	H. Tierney
A. Mannino	A. Turner

Additional Distribution

O.R. Barker	C.A. Miglierina
J.P. Chiasson	J. McKissick, Jr.
J.A. Johnson	R.E. Ostrowski
C.H. LeVine	R.A. Price
J.M. MacDonough	B.D. Roberts
W.G. Materne	J.A. Stephens
E.B. McCrohan	D.D. Ward
C.C. McFarling	

Accession For	
NTIS GMA&I	<input checked="" type="checkbox"/>
DDC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	<input type="checkbox"/>
By _____	
Distribution/	
Availability Codes	
Dist.	Avail and/or Special
A	

TABLE OF CONTENTS

	<u>Page</u>
Introduction	1
What is an SDN	1
SDN Audits	4
Experience and Results	5
Summary	5

LIST OF ILLUSTRATIONS

<u>Figure</u>	<u>Title</u>	<u>Page</u>
1	SDN Sections	1
2	Overview Status Sheets	2
3	Detailed Status Sheet	2
4	SPR and SCO Logs	3
5	Code Listing Deficiencies	5

Key Words: Software, Software Audit, Software Maintainability, Software Quality, Software Reliability.

Introduction

A Software Development Notebook (SDN) technique is described in this paper. Significant benefits have resulted from the use of this technique over the past three years. Software is developed in a more orderly and disciplined manner. The emerging software product is visible and auditable. Deficiencies are identified and corrected earlier. Documentation evolves in the required format. Software and documentation are more consistent. Current schedule status is available. The net results are improved software engineering discipline and an improved software product.

A general description of the SDN technique was first presented in a prior paper. (Ref. 1) The subsequent interest shown by both software engineering and software R&QA professionals motivated us to prepare this paper devoted to the SDN and SDN audits.

The question "What is an SDN?" is answered in the next section of this paper. This is followed by a brief description of how SDN's are established and maintained. SDN audit planning and conduct are then addressed. In conclusion, lessons learned from our use of the SDN technique are presented.

What is an SDN?

To the programmer, the SDN is his day-to-day working notebook. To Computer Software R&QA, the SDN is a window through which the software process and the emerging software product are viewed.

A Software Development Notebook is a loose-leaf notebook which provides a common collection point for all current information relating to a computer program component (CPC)*. The sections of the SDN are identified in Figure 1. The order of these sections is compatible with the Software Development Process documented in reference 1.

The content and format requirements for each section of the SDN are defined in detail in the Software Standards and Procedures Manual. A summary of each SDN section follows.

*A computer program is partitioned into computer program components which are further partitioned into routines.

Section 0 - Status Sheets

Section 0 of the SDN contains status sheets which document schedule, completion, approval and audit status. The first status sheet provides an overview for the CPC. Detailed status sheets follow for sections 2 through 6 of the SDN. These status sheets are useful in combating the "90% complete" syndrome. (Ref. 2)

An overview status sheet is shown in Figure 2. This status sheet provides visibility of the development progress and audit status for a CPC. Figure 3 shows a typical detailed status sheet which provides similar visibility to the routine level.

Section 1 - Requirements

Computer Program Performance Specification (CPPS) requirements currently allocated to this CPC are listed in Section 1. The listing includes CPPS paragraph number, requirement number, a summary statement of the requirement and the names of the routines which implement and satisfy the requirement.

Early in the software development cycle, some CPPS requirements will be missing or incomplete. Any requirements assumptions made so that software development can proceed are documented in this section of the SDN. Copies of letters and memos which relate to requirements issues are also included.

This section serves as a constant reminder of current requirements as the CPC is designed and tested.

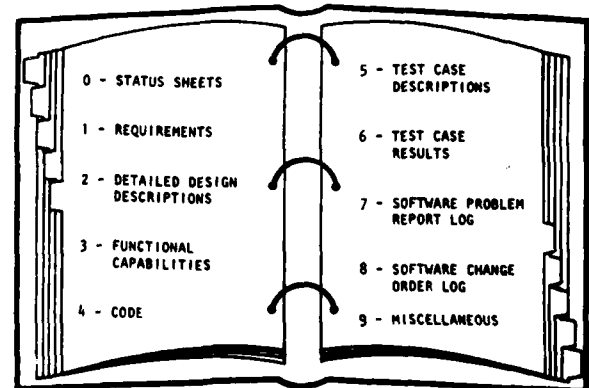


Figure 1. SDN Sections

Section 2 - Detailed Design Descriptions

This section contains the current detailed design of the CPC. It typically includes subsections for a CPC overview, a data base description and the description of each routine. Significant cost savings result from tailoring the format and content of this section to meet final documentation requirements.

The CPC Overview Subsection provides a brief general description of the functions performed by the CPC and includes a hierarchical diagram showing its structure. This subsection also describes all external CPC interfaces.

The CPC Data Base Subsection contains a complete description of the local variables used by the CPC and identifies the routines which set or use each variable. The format and content of tables used in this subsection and data base naming conventions are defined in detail in the Software Standards and Procedures Manual.

The Routine Detailed Design Subsections provide a description of each routine. The description includes functions performed, enablement criteria, inputs, outputs, a processing description and a routine data base description.

Section 3 - Functional Capabilities

Functional capabilities of the software design are listed in this section and traced to CPPS requirements. They provide the basis for developing the CPC and routine test cases described later.

Separate functional capability lists are included in this section of the SDN for the CPC and for each routine. A traceability matrix relating CPPS require-

ments to CPC and routine level functional capabilities is also included.

Section 4 - Code

A current code listing for each routine is kept in a numbered binder at a central location. This section of the SDN contains a table which provides a cross reference between routines and binder numbers.

Section 5 - Test Case Descriptions

In this section CPC and routine level test cases are described and traced to functional capabilities of the software.

The description of each test case includes its identification number and purpose, inputs, support software requirements and criteria for successful completion. Separate traceability matrices are also included for the CPC and each of its routines relating functional capability identification numbers to test case identification numbers.

Section 6 - Test Case Results

Like code listings, hard copy results of tests are kept in numbered binders at a central location. These results are clearly marked to show that the criteria for successful completion have been met. This section of the SDN contains a table which provides a cross reference between test case numbers and binder numbers.

Completion and approval of Section 6 indicate that the SDN has served its major purpose and the CPC is ready for integration testing. The SDN will continue to be used for software modification and change control.

SOFTWARE DEVELOPMENT NOTEBOOK STATUS SHEET					
CPC NAME <u>RADAR CONTROL</u>					
CPC NAME <u>RCMNT</u> RESPONSIBLE INDIVIDUAL <u>R. JONES</u>					
SECTION NUMBER	CONTENT	DUE DATE	DATE COMPLETE	CHIEF PROGRAMMER APPROVAL	RQA AUDIT
1	Requirements	4/15	4/16	Jan	①②③
2	Detailed Design Descriptions	5/25	5/24	Jan	①②③
3	Functional Capabilities	6/10	6/5	Jan	①⑤
4	Code	4/25			①④
5	Test Case Descriptions	6/25			①
6	Test Case Results	7/10			①
7	Software Problem Report Log	N/A	N/A	N/A	①
8	Software Change Order Log	N/A	N/A	N/A	①
9	Miscellaneous	N/A	N/A	N/A	N/A

① MAR. 20, 1978 B.J.P. ④ JUNE 20, 1978 CDR.
 ② APR. 25, 1978 B.J.P.
 ③ MAY 29, 1978 B.J.P.

Figure 2. Overview Status Sheet

DETAILED DESIGN DESCRIPTION STATUS SHEET						
CPC NAME <u>RCMNT</u>						
SUBSECTION NUMBER	DESCRIPTION	DATE DUE	DATE COMPLETE	RESPON. PROG.	CHIEF PROG. APPROVAL	RQA AUDIT
2a	CPC Overview	5/1	5/5	RJ	Jan	①
2b	CPC Data Base	5/25	5/24	JM	Jan	②
2c 0	Control Routine VPMTRKX	4/20	4/20	RJ	Jan	③
2c 1	Routine VPMHITZ	4/26	4/20	CCM	Jan	②
2c 2	Routine VPMTRMX	5/12	5/14	CCM	Jan	③
2c 3	Routine VPMSTART	5/25	5/21	CCM	Jan	
2c 4	Routine VPMSTOPX	4/26	4/23	BDR	Jan	②
2c 5	Routine VPMSENR	5/12	5/10	BDR	Jan	③
2c 6	Routine VPMSCHE	5/25	5/23	BDR	Jan	
2c 7	Routine VPMOSPLY	4/26	4/26	JM	Jan	②③
2c 8	Routine VPMLISTX	5/25	5/22	JM	Jan	③
2c 9	Routine	—	—	—	—	—
2c 10	Routine	—	—	—	—	—

② APR. 25, 1978 B.J.P.
 ③ MAY 20, 1978 B.J.P.

Figure 3. Detailed Status Sheet

Section 7 - Software Problem Report Log

The identification and resolution of software problems improve the reliability and quality of the final software product. This section and Section 8 of the SDN provide visibility and control of these activities.

An SPR log and a copy of all Software Problem Reports (SPR) affecting this CPC are contained in Section 7. SPR's define and document problems identified during integration and acceptance testing. They also document the test conditions under which the problems occurred. Software problem reporting and the SPR we use are described in reference 1. The SPR log we use is shown in Figure 4.

Section 8 - Software Change Order Log

Section 8 contains a Software Change Order (SCO) log and a copy of all SCO's affecting this CPC. SCO's document and control changes to code. SCO's are prepared in response to approved CPPS requirements changes and Software Problem Reports. The SCO log we use is shown in Figure 4.

Section 9 - Miscellaneous

This section contains material relative to the design, code and test of the CPC which the individual responsible for the CPC feels is appropriate for present or future use. This material typically includes working notes, reference tables, technical reports, memos and correspondence. Copies of the most recent SDN Audit Report and responses to this report are also included.

A general description of the Software Development Notebook has been provided. It is important that the detailed format, content and organization of the SDN for a project be specified in the Software Standards and Procedures Manual early in the Software Development Process. It is equally important that this manual specify the procedures and responsibilities for establishing and maintaining the SDN.

Establishing and Maintaining the SDN

An SDN is established for each computer program

component (CPC) at the start of the Detail Design Phase of the Software Development Process. It is maintained current throughout the remainder of the development process.

The physical establishment of an SDN is a clerical procedure performed by the project librarian under the direction of the Chief Programmer. It entails marking a loose-leaf notebook with the project and CPC name, inserting labeled tab dividers for each section of the SDN and placing status sheets in Section 0, the CPC requirements list in Section 1 and log forms in Sections 7 and 8. The SDN is then provided to the individual responsible for the CPC.

Based on a completion date provided by the Chief Programmer, the individual responsible for the CPC enters due dates for each section on the overview status sheet. Compatible due dates for each routine are later entered on the detailed status sheets by the responsible programmer. All schedules are reviewed and approved by the Chief Programmer. As a subsection or section of the SDN is completed, the completion date is entered on the appropriate status sheet. After completing his review, the Chief Programmer initials the status sheet to indicate approval. Section 0 of the SDN therefore provides current CPC schedule and approval status at all times.

The individual responsible for the CPC augments the requirements list in Section 1 to show allocations to the routine level and updates the list as these allocations or requirements change. He also enters and logs Software Problem Reports in Section 7 when they are issued and resolved, and enters and logs Software Change Orders in Section 8 when they are approved and implemented. The inclusion and purging of material in Section 9 is also the responsibility of this individual.

The remaining sections of the SDN are the responsibility of the computer programmers. They describe the software design and design capabilities in Sections 2 and 3, develop the code and test cases in Sections 4 and 5 and document test results in Section 6. Keeping the SDN current, readable and compliant with the software standards is an on-going computer programmer responsibility.

SOFTWARE PROBLEM REPORT LOG					
CPC NAME _____					
SPR #	DATE RECEIVED	DESCRIPTION	DATE DUE	DATE COMPLETE	NOTES

SOFTWARE CHANGE ORDER LOG						
CPC NAME _____						
SCO #	DATE RECEIVED	DESCRIPTION	DATE DUE	DATE COMPLETE	RESPON. PROG.	NOTES

Figure 4. SPR and SCO Logs

The importance of maintaining Software Development Notebooks which are current and compliant with software standards cannot be overemphasized. Well maintained SDN's minimize later integration problems by providing design and interface visibility to other programmers. A costly and time consuming documentation cycle is also avoided by evolving final software documentation in the SDN. The procedures described above have been used on several major projects over the last three years. The result has been improved in-house and subcontractor software and documentation.

SDN Audits

In addition to their value as a software development technique, SDN's are a cost effective control and assurance technique. They provide visibility into the Software Development Process and the emerging software product. SDN's facilitate audit activity by providing all needed information in one location. Our SDN audits are responsive to the requirements of MIL-S-52779. (Ref.3)

The following principles guide SDN audit planning:

- SDN audits must be an element of a comprehensive Software Quality Assurance Program (Ref. 1) which should be documented as an integral part of a Software Development Plan.
- SDN audits should be conducted by individuals who are organizationally independent of the software developers.
- SDN audits should begin early and continue through the entire development process.
- SDN audits should be professionally planned, conducted, and documented, and corrective action should be followed.

Our planning recognizes the need to shift audit emphasis for different software development activities. SDN audits and audit criteria during design, code and test are described next.

SDN Audits During Design

The initial SDN audit is conducted early in the Detail Design Phase to verify that SDN's have been established for all identified computer program components (CPC's) and are organized in accordance with software standards. Sections 0 and 1 are checked at this time to assure that a schedule and a requirements list have been established.

Later audits during this phase confirm that Section 0 includes detailed schedules and that completion and approval status is current. The requirements list in Section 1 is checked to assure that it is up-to-date and shows requirement allocations to the routine level.

CPC routine and data base design descriptions in Section 2 are audited for compliance with software standards and for consistency. Audit criteria include format, content and nomenclature standards. Flowcharts or functional flows are checked for allowable constructs, symbology and structure. Data tables are checked for required format and content. Technical and nomenclature consistency throughout the design description are emphasized during these audits.

SDN Audits During Code

SDN audit emphasis shifts when code becomes

available. Computer Software R&QA auditors use detailed status sheets in Section 0 to determine code availability and the table in Section 4 to locate code listings. These listings are checked for compliance with coding standards such as structure, annotation and routine size. Agreement between the code listings and the approved design descriptions is also verified. Instruction sequences and data usage are both checked.

SDN Audits During Test

SDN audits during routine and CPC testing focus on functional capabilities in Section 3, test case descriptions in Section 5 and test case results in Section 6. The auditor first checks that functional capabilities have been identified and traced to requirements. Test case descriptions and associated traceability matrices are then audited to assure that all functional capabilities will be tested. Finally, test results are audited to determine that all tests were conducted and satisfactorily completed.

The emphasis of SDN audits during integration and acceptance testing is to assure that identified problems are resolved and approved changes are implemented. These audits address the Software Problem Report (SPR) logs in Section 7, the Software Change Order (SCO) logs in Section 8 and resulting changes in other sections of the SDN. SPR and SCO logs are checked against configuration control board records to verify that the logs are complete. The inclusion of all logged SPR's and SCO's is then checked. These documents are used to verify that problems have been resolved and approved changes have been implemented.

SDN Audit Planning, Procedures and Documentation

The approved Software Development Plan includes a section describing the Software Quality Assurance Program. SDN audit scope, frequency, sample selection and documentation are defined in this section. An approved implementation plan is required prior to each audit which defines the audit objectives, criteria, sample, schedule and audit team personnel. Audit objectives and audit criteria are based on the current software development phase, the results of prior audits and recommendations from Chief Programmers and managers. The audit sample is selected to include work from each computer programmer. Experience has shown that routines which are complex or significantly behind schedule should also be selected for audit.

Since the SDN is well defined and self-contained, an independent audit can be conducted without involving the software developers. This approach results in SDN audits which are objective and non-disruptive. A previously prepared checklist is completed for each SDN audited. The auditor identifies the subsection of the SDN being reviewed and then describes any deficiencies. The audit team leader uses these completed checklists to prepare the audit report. This report documents audit findings and associated recommendations. Prior to issuing the report, the audit team leader validates the technical accuracy of audit findings with the Chief Programmer.

Response forms are provided to the Chief Programmer on which he defines corrective actions and schedules completion dates. Computer Software R&QA uses these completed forms to verify that corrective action is defined and implemented.

Problems have been identified by SDN audits during all phases of the Software Development Process. Cost savings were realized because problems identified

early in the process are far less difficult and less costly to correct. (Ref. 4) Moreover, some of these problems, if not found and corrected, would have had major impact on software quality, reliability and maintainability.

Experience and Results

The SDN technique described in this paper has been applied on major projects over the past three years to provide control and assurance for both in-house and subcontractor developed software. The SDN audit technique has led to the acceptance of problem identification and correction activity as integral parts of the Software Development Process. Significant qualitative and quantitative results from this experience are presented below.

The most important prerequisite for successful application of the SDN technique is the Software Standards and Procedures Manual. This manual should unambiguously specify the details of SDN format, content and organization. The SDN should be easy to use and maintain. It should be noted that improved software standards result from the use of standards as audit criteria.

The support of functional and project management is needed when the SDN technique is instituted. Resistance from some software developers should be anticipated. The technique will be accepted by these developers when they experience its benefits.

We have found that a Computer Software R&QA engineer assigned to a software project provides valuable continuity and can effectively serve as the leader of the independent SDN audit team. He must be able to read code.

Our in-house and subcontractor audit findings have led us to some expected and some surprising observations. Among these are the following:

- The major problem identified by SDN audits was lack of consistency between design descriptions and code listings.
- The quality of a programmer's work on a project remains relatively constant and varies widely from programmer to programmer.
- Frequent requirements changes, schedule difficulty and personnel turnover can adversely impact software quality.

These observations should be considered in selecting audit samples.

Analysis of 1175 deficiencies identified during monthly SDN code audits provided the quantitative results which follow. Most of the deficiencies (842 or 72%) were inconsistencies between code listings and design descriptions. An example of this type of deficiency is inconsistency between the code annotation and the design description. The remaining deficiencies (333 or 28%) were non-compliance of code listings with software standards. An example of this type of deficiency is a routine prologue which lacks required content.

As Figure 5 illustrates, commentary deficiencies predominate in both categories. The majority of these deficiencies were in the documentation rather than the code. Correcting them improved software maintainability.

Summary

The Software Development Notebook technique is being used in the development of computer software. The success of the technique depends upon well-defined software standards and management support.

An SDN is a loose-leaf notebook which provides a common collection point for all current information relating to a computer program component (CPC). It includes schedule, requirements, design, code, test and change control information. The establishment of an SDN is a clerical procedure. It is maintained by the software developers who must keep it current.

SDN's are a cost effective control and assurance technique. They provide visibility and facilitate independent audit activity. The emphasis of audit activity shifts with the phases of the Software Development Process. SDN audits must be professionally planned, conducted, documented and corrective action must be followed.

We have used this SDN technique over the past three years for in-house and subcontractor developed software. SDN's are easy to use and to audit. Our audits are led by an independent Computer Software R&QA engineer assigned to the project. The major problem identified by SDN audits was lack of consistency between design descriptions and code listings. Improved software maintainability resulted from the subsequent corrective actions.

The "bottom line" is an improved software product.

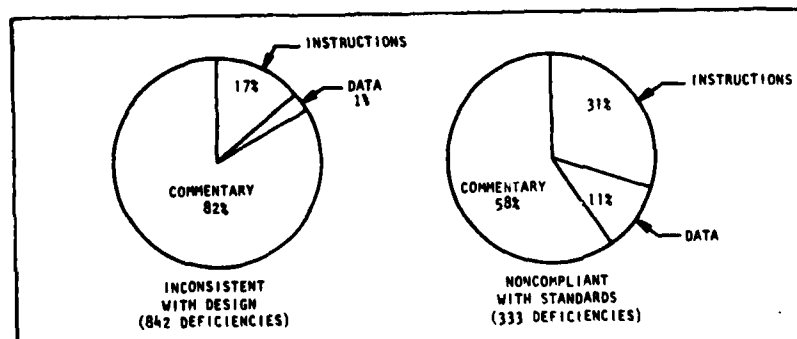


Figure 5. Code Listing Deficiencies

References

1. J. McKissick and R. A. Price, "Quality Control of Computer Software", 1977 ASQC Technical Conference Transactions, May 1977.
2. F. S. Ingrassia, "Combating the '90% Complete' Syndrome", Datamation, January 1978.
3. MIL-S-52779 (AD), "Military Specification, Software Quality Assurance Program Requirements", April 1974.
4. B. C. DeRoze, "Software Management Within the Department of Defense", NSIA Conference on Software Quality-Reliability Transactions, March 1977.

Progress Is Our Most Important Product

GENERAL  ELECTRIC